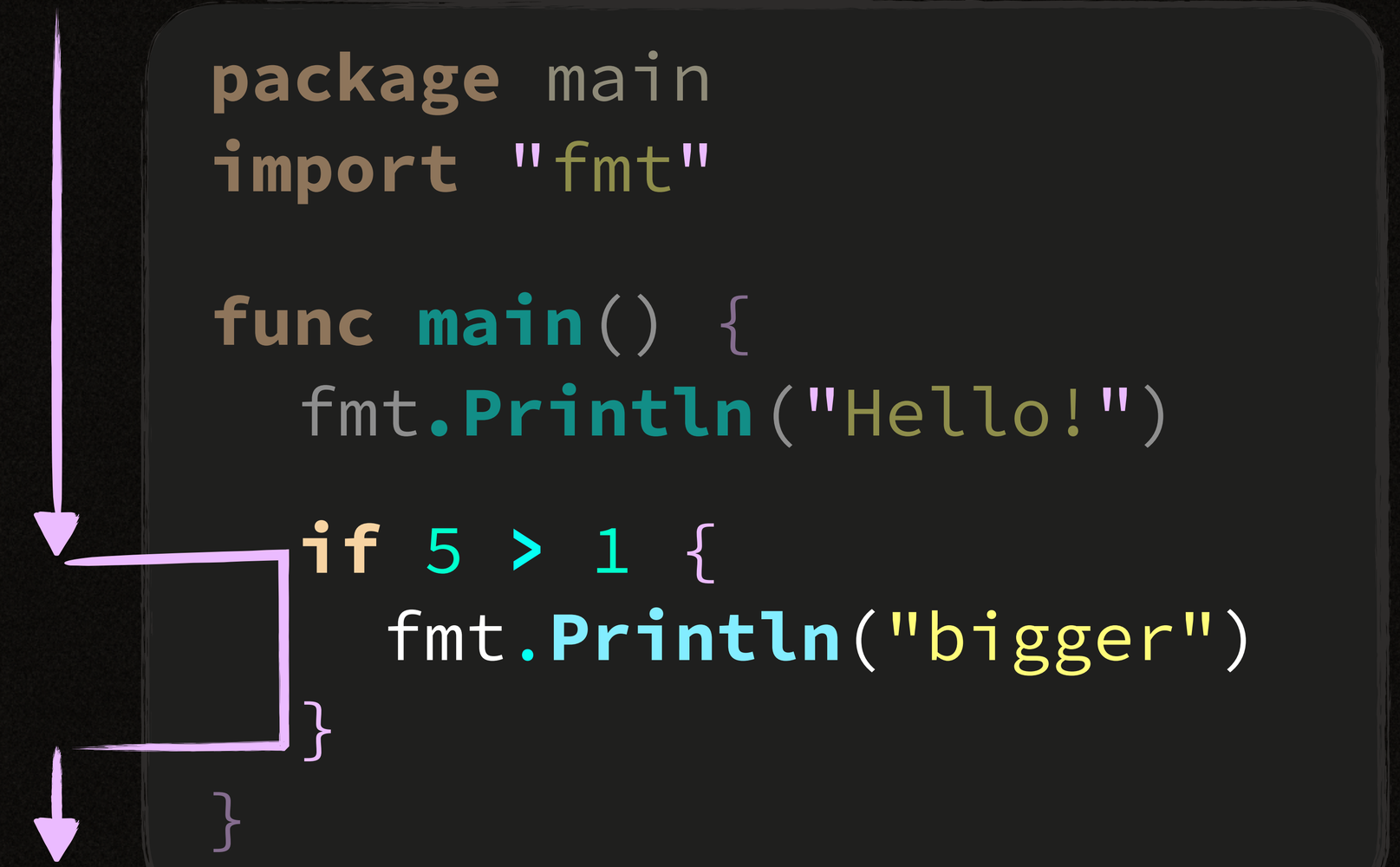
PART III

Fundamentals

Control Flow

```go
package main
import "fmt"

func main() {
   fmt.Println("Hello!")
}
```

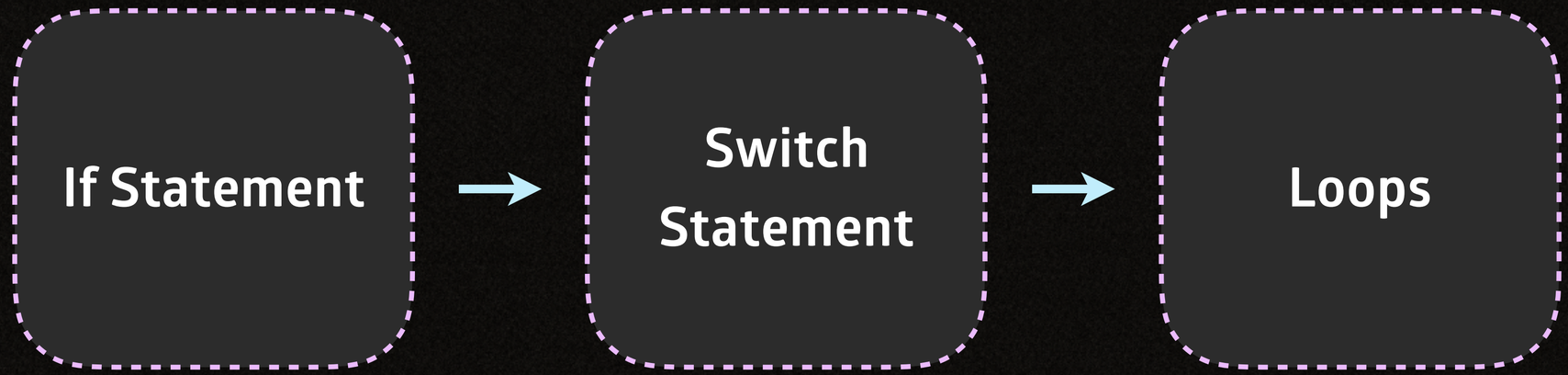


```go
package main
import "fmt"

func main() {
    fmt.Println("Hello!")

    if 5 > 1 {
        fmt.Println("bigger")
    }
}
```

PART III

Fundamentals

Control Flow



IF STATEMENT

*Time to learn how to **control** the **execution** flow*

Boolean Operators → **If Statement** → **Error Handling** → **Simple Statement**

Comparison and,
Logical Operators

Only for
—beginners—

Refactor:
Feet to Meters

Project:
PassMe

nil value
error value

Refactor:
Feet to Meters

Famous
Shadowing
Gotcha

BOOLEAN OPERATORS

You're going to learn about boolean operators

Comparison Operators → Logical Operators

OPTIONAL

for experienced programmers

summary

comparison operators

equality operators

ordering operators

equality operators

```
true == true   →   true
true != true   →   false
```

ordering operators

```
1 < 2   →   true

2 > 5   →   false

2 >= 2  →   true

3 <= 2  →   false
```

assignability and comparison

```
i, f := 15, 5.0
```

```
i > f
```

```
i > int(f)
```

a logical operator

combines

bool expressions

and yields

a bool value

logical operators

Logical AND Operator

```
true && true      → true
```

Logical OR Operator

```
false || false    → false
```

Logical NOT Operator

```
!false            → true
!!false           → false
```

short-circuit

```
false && true
```

runs

skips

```
true || false
```

runs

skips

IF STATEMENT

Congrats!

Boolean Operators ✓ → If Statement → Error Handling → Simple Statement

summary

if statement doesn't require parentheses

```
if (true) {...}   ✗
```

↓

```
if true   {...}   ✓
```

condition expressions can only be bool

```
if 1 {...}
```

```
if true {...}
```

directly use bools

```
if true == true {...}
```

```
if true {...}
```

if statement

```
if true {
} else if true {
} else if true {
} else if true {
} else {
}
```

IF STATEMENT

Congrats!

Boolean Operators ✓ → If Statement ✓ → Error Handling → Simple Statement

ERROR HANDLING BASICS

You're going to learn how to handle the errors

Why?

Not everything goes according to the plan

Things can fail

- Network problems
- File access problems
- User input problems
- ...

Error handling is crucial

And it is part of your programming job

summary

nil

```
if err != nil {
  // ❌ error
  // quit
}
// 👍 success
```

error

```
if err != nil {
  // ❌ error
  // quit
}
// 👍 success
```

immediately handle the errors

```
if err != nil {
  // ❌ error
  // quit
}
// 👍 success
```

there is no throw/catch

```
if err != nil {
  // ❌ error
  // quit
}
// 👍 success
```

IF STATEMENT

Congrats!

Boolean Operators ✓ → If Statement ✓ → Error Handling ✓ → Simple Statement

SHORT IF

You're going to learn how to use Simple Statements with an if statement

summary

simple statement

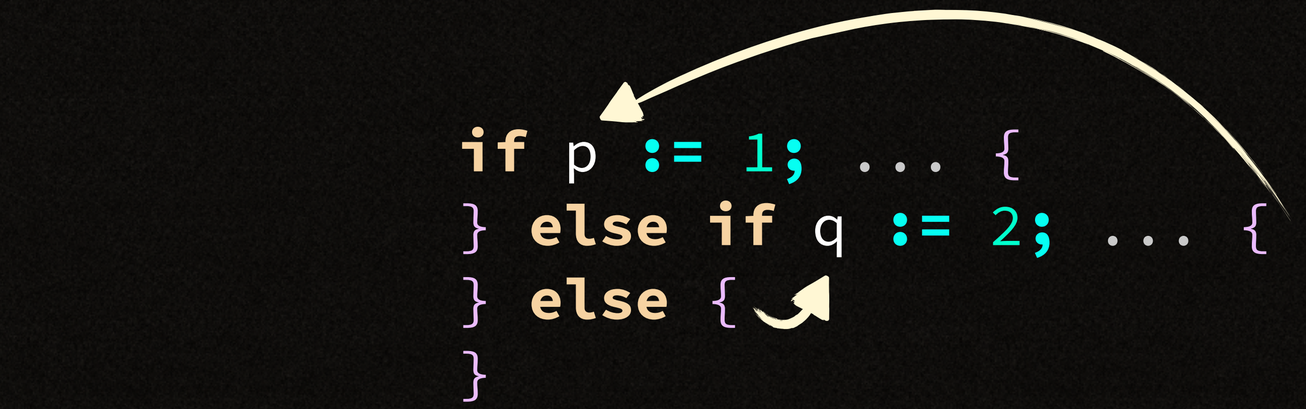
(as known as the short statement)

```
err := do()
if err != nil {}
```



```
if err := do(); err != nil {}
```

scopes



shadowing gotcha

```
err := do()
```

```
if err := run(); err != nil {}
```

IF STATEMENT

Congrats!

Boolean Operators ✓ → If Statement ✓ → Error Handling ✓ → Simple Statement ✓

PART III

Fundamentals

Control Flow